BINARY-CODED, AUTO-ADDRESSING SYSTEM AND METHOD

BACKGROUND OF THE INVENTION

Field of the Invention

The present invention is related to the field of addressing multiple devices on a multiplexing bus and, more particularly, to a binary-coded method for auto-addressing a plurality of devices within a network by their positions in the network.

Description of the Related Art

Known methods of auto-addressing multiple devices on a multiplexing bus include memory-flashing and pin-coding. However, these approaches are not cost-effective, and require unique components such as individually pre-addressing devices or pre-addressing connectors. Therefore, a need exists for an auto-addressing method that is compatible with typical multiplexing bus systems and able to address devices simply and cost-effectively.

SUMMARY OF THE INVENTION

In view of the foregoing, one object of the present invention is to overcome the difficulties of auto-addressing systems and methods that require special components by providing a system and method that is able to auto-address a plurality of devices based upon their position in the network.

Another object of the present invention is to provide an auto-addressing method for use on a multiplexing bus system with a master and sequentially-arranged slave devices, in which the slave devices have identical electronic components and execute the same logical flow.

A further object of the present invention is to provide an auto-addressing method executable via either a microprocessor alone or through a combination of software and hardware.

An additional object of the present invention is to provide an auto-addressing method using a sequence of evaluations based on the current bus-in signal and all previous bus-out signals to determine an address for each device on the bus.

Yet another object of the present invention is to provide a digital auto-addressing method that is not affected by variations in supply voltage, ambient temperature, sense resistance values, etc., and which converges quickly, needing only $\log_2(n)$ measurements for n devices.

It is yet another object of the invention to provide an auto-addressing method which can be cost-effectively incorporated into existing multiplexing bus systems to efficiently decode all addresses without specialized components.

In accordance with this and other objects, the present invention is directed to a method for auto-addressing devices on a multiplexing bus having a master control module and a plurality of

slave devices arranged in series, with each slave device having an address register, a bus in and a bus out. A first bus signal having a first state (either high or low) is output from the master control module to the first slave device and then sequentially passed to each subsequent slave device in the series. Upon receipt of the bus signal at the bus in, each slave device determines the content of its respective address register and inverts the bus in signal to an inverted bus out signal only if the value of its respective address register content is zero. The address register content of each slave device is updated with the respective bus out value for that device after the first measurement, and the process is repeated with a second bus signal, again having the same first state, being output from the master control module and passed through the slave devices in series. When $\log_2(n)$ measurements have been taken, with n being the number of slave devices daisy-chained together, the process is complete and the address of each of the slave devices will have been determined.

These and other objects of the invention, as well as many of the intended advantages thereof, will become more readily apparent when reference is made to the following description taken in conjunction with the accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 shows a typical arrangement of serially-arranged devices in a multiplexing bus system;

Figure 2 is a representative electrical schematic of the auto-addressing scheme of a slave device within a multiplexing bus system according to the present invention; and

Figure 3 is a flow chart of the addressing method according to the present invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

In describing a preferred embodiment of the invention illustrated in the drawings, specific terminology will be resorted to for the sake of clarity. However, the invention is not intended to be limited to the specific terms so selected, and it is to be understood that each specific term includes all technical equivalents which operate in a similar manner to accomplish a similar purpose.

Although only one preferred embodiment of the invention is explained in detail, it is to be understood that the embodiment is given by way of illustration only. It is not intended that the invention be limited in its scope to the details of construction and arrangement of components set forth in the following description or illustrated in the drawings. Also, in describing the preferred embodiments, specific terminology will be resorted to

for the sake of clarity. It is to be understood that each specific term includes all technical equivalents which operate in a similar manner to accomplish a similar purpose.

The auto-addressing method according to the present invention works on a typical multiplexing bus system, such as one used in vehicular architecture and representatively depicted in Figure 1. Such a bus system includes a master control module 110 and a plurality of slave devices 112, 114, 116 connected in series over a three-wire cable 20, in which a first wire 22 carries the power supply voltage, a second wire 24 provides ground, and a third wire 26 is the bus line. Slave devices in this bus arrangement can be spliced to the cable 20 and are controlled by the master control module 110. Each slave device has an address register with a length of $\log_2(n)$ bits, with n representing the maximum number of allowed devices on the bus.

The bus line 26 enters a bus input side 30 of the slave devices 112, 114, 116 and exits a bus output side 32 thereof, sequentially, such that the bus output 32a of the first slave device 112 is the bus input 30b of the second slave device 114, and the bus output 32b of the second slave device 114 is the bus input 30c of the next slave device 116, and so forth. The bus input 30a of the first slave device 112 is directly tied to the master control module 110.

An electrical schematic representative of the auto-addressing circuitry of each slave device is illustrated in Figure 2. While the execution according to the present invention can be implemented either in software via a microprocessor or by a combination of software and simple hardware, the implementation illustrated in Figure 2 represents a software/hardware combination.

As shown in Figure 2, the slave device includes a microprocessor 40 with an address register 42 and a measurement counter 41, and an address logic block 44. Both the microprocessor and the logic block 44 receive the bus in signal 26a from the bus line 26 via the bus input 30.

Upon receipt of the bus in signal 26a through the bus input 30, the microprocessor 40 checks the address register 42. If the value stored in the address register is zero, the microprocessor enables the inverter signal 46; conversely, if the value stored in the address register 42 is not zero, i.e., at least one of the $\log_2(n)$ bits is a "1", the microprocessor disables the inverter signal 46.

The logic block 44 receives the inverter signal 46 and the bus in signal 26a, and either inverts the bus in signal 26a or passes such signal through unchanged, depending upon the inverter signal. According to a preferred embodiment, the logic block is configured to perform a logic XOR operation of the bus in and

inverter signals, inverting the bus in signal only if both inputs are high or if both inputs are low.

The logic block 44 outputs the bus out signal 26b which is output via the bus output 32 to the bus line 26 for input to the bus input 30 of the next slave device. In addition, the bus out signal 26b is fed back to the microprocessor 40 to enable the microprocessor to update the next bit in the address register 42 with the most recent output state, as will be explained in greater detail subsequently herein.

As an alternative embodiment, the slave device circuitry may be embodied entirely in software in which case the logic block is eliminated. In such a software embodiment, the bus in signal 26a is received by the microprocessor 40 as shown in Figure 2, but the microprocessor outputs the bus out signal directly to the bus line 26. In this embodiment, there is no feed back of the bus out signal to the microprocessor as the microprocessor already knows the latest output state.

The auto-addressing sequence according to the present invention is summarized in Figure 3 and is initiated by the master control module. To begin the auto-addressing sequence or mode, step 200, the master control module 110 sends a command signal to each of the slave devices 112, 114, 116, and pulls the bus signal on bus line 26 either low or high. In response to receiving the command signal, each slave device initializes its own address

register to zero, step 202. Thereafter, throughout the measurements necessary to complete the address-decoding sequence, the master control module holds the bus signal at the same state, whether low or high.

All slave devices make $\log_2(n)$ measurements, where n is the maximum number of devices allowed on the bus. When all slave devices receive the command from the master control module to auto-address and subsequently zero out their $\log_2(n)$ bit address registers, the value, i, of the measurement counter 41 within the microprocessor 40 is also set to zero and the measurement counter is initialized such that the total number of measurements, i_{max}, is equal to $\log_2(n)$, step 202. Since n is the maximum number of devices allowed on the bus, a bus having sixteen devices requires four measurements to determine the addresses of each of the devices such that i_{max} is set to four; a bus having eight devices requires three measurements such that i_{max} is set to three, and so on.

Each of the subsequent measurements follows the same procedure. Particularly, in response to receiving the bus in signal 26a, the microprocessor 40 in each device examines the content of its respective address register 42 and enables the inverter signal 46 only if the content of the register is equal to zero. At the first measurement, all of the registers have a content of zero due to the initialization performed in response to the command signal.

For each slave device, if the value stored in the device's register is zero, i.e., all $\log_2(n)$ bits are "0", step 204, the inverter signal 46 is enabled and the bus output of that device is the inverse of its bus input, step 206. If, on the other hand, the address stored in the register is not zero, i.e., if any of its $\log_2(n)$ bits is a "1", step 204, then the inverter signal is disabled and a transmission gate or pass-through within the logic block 44 is enabled; hence, the bus output of a slave device in which the content of the register 42 is not zero is of the same state as its bus input, step 208.

At the end of each measurement, the address register is updated to reflect the output state resulting from the last measurement, step 210, with the least significant bit (LSB) of the address register corresponding to the first measurement, the next bit corresponding to the second measurement, and so on. The value of the measurement counter, i, is then compared with the total number of measurements, i_{max}, to see if $i = i_{max}$, step 212. If i does not equal i_{max}, the measurement counter is incremented by one, step 214, and the next measurement is initiated. Conversely, if $i = i_{max}$, the measurement sequence has been completed, step 216.

For purposes of illustration, the measurement sequence will now be examined in greater detail in an embodiment in which the bus signal from the master control module is held high, while

noting that the invention works equally well with the bus signal held low.

Following initialization to begin the auto-addressing sequence, a high bus signal is sent to the devices which then conduct a first measurement to determine if the content of their respective address registers is zero, step 204. More specifically, the first device 112 (closest to the master) reads the bus signal which is high, i.e., a "1". Since the content of the address register of the first device is zero from the initialization, the bus output 32a of the first device 112 is the inverse of the bus input 30a, step 206, and since the bus input is a "1", the bus output 32a is a "0". This bus output is the bus input 30b of the second device 114.

Correspondingly, because the content of its address register is also zero, the second device 114, in turn, inverts its bus input signal 30b, so that its bus output 32b is a "1". Thus, since the value of the register of each device is set to zero at initialization, each device inverts its incoming signal so that, at the first measurement, the output of each device is the inverse of the output of the device before it, step 206. Each device then stores its own output state in the LSB of its address register (A_0), step 210. Table 1 summarizes the input and output states of all of the slave devices on the bus at the first measurement, with

Device 0 being the first device immediately following the master control module.

TABLE I

	First Measurement Master BUS Signal "1"	
	BUS_{IN}	$A_0=BUS_{OUT}$
Device 0	1	0
Device 1	0	1
Device 2	1	0
Device 3	0	1
Device 4	1	0
Device 5	0	1
Device 6	1	0
Device 7	0	1
Device 8	1	0
Device 9	0	1
Device 10	1	0
Device 11	0	1
Device 12	1	0
Device 13	0	1
Device 14	1	0
Device 15	0	1

The value i in the measurement counter is then compared with the maximum value for i, i_{max}, step 212, to determine if the measurement sequence has been completed. If i does not equal i_{max}, the measurement counter is incremented by one, step 214, and the process continues with a next measurement.

In the second measurement, the master control module still holds its bus signal high, i.e., a "1". The first device 112 reads this bus signal directly and determines if the content of its register is equal to zero, i.e., all $\log_2(n)$ bits are "0", step 204. Since the content of the LSB (A_0) of the address register of

the first device is a "0" from Table 1, the bus output 32a is the inverse of the bus input 30a, step 206, and is a "0". This output is the bus input 30b for device 114.

Device 114 then determines if the content of its register is equal to zero. Since the content of the LSB (A_0) of the address register of the second device is not zero, step 204, instead having a "1" in the LSB of its address register, the second device passes its bus input 30b through without inverting it, step 208; thus the bus output 32b of the second device 114 is a "0".

Device 116 then determines if the content of its register is equal to zero. Since the content of the LSB (A_0) of the address register of the third device has a "0" in the LSB, upon receiving bus input 30c as a "0", device 116 inverts the bus signal and has a bus output 32c of "1", step 206. This process continues in a similar fashion for each of the devices in the network.

Following the second measurement, each device again stores its own most recent output state in the next bit of its address register (A_1), step 210. A summary of the input and output states of all the devices on the bus after two measurements is presented in Table 2.

TABLE II

	Second Measurement Master BUS Signal "1"	
	BUS_{IN}	$A_1=BUS_{OUT}$
Device 0	1	0
Device 1	0	0
Device 2	0	1
Device 3	1	1
Device 4	1	0
Device 5	0	0
Device 6	0	1
Device 7	1	1
Device 8	1	0
Device 9	0	0
Device 10	0	1
Device 11	1	1
Device 12	1	0
Device 13	0	0
Device 14	0	1
Device 15	1	1

The value i in the measurement counter is then compared with the maximum value for i, i_{max}, step 212, to determine if the measurement sequence has been completed. If i does not equal i_{max}, the measurement counter is incremented by one, step 214, and the process continues with a next measurement.

In the third measurement, the master control module still holds its bus signal high, i.e., a "1". The first device 112 reads this bus signal directly and determines if the content of its register is equal to zero. Since the content of the first two bits (A_1, A_0) of the address register of the first device is "00" from Tables 1 and 2, step 204, the bus output 32a of the first device is the inverse of the bus input 30a, step 206, and is a "0". This output is the bus input 30b for device 114.

Device 114 then determines if the content of its register is equal to zero. Since the content of the register of the second device is not zero, instead having a "01" in its address register's A_1, A_0 bits, the second device passes its bus input 30b through without inverting it, step 208; thus the bus output 32b of the second device 114 is a "0".

Device 116 then determines if the content of its register is equal to zero. However, because the third device 116 has a "10" in its register for the two previous measurements, the third device does not enable its inverter but passes its bus input 30c through without inverting it, step 208; thus the bus output 32c of the third device 116 is a "0".

This process continues in a similar fashion for each of the devices in the network. Following the third measurement, each device again stores its own most recent output state in the next bit of its address register (A_2), step 210. A summary of the input and output states of all the devices on the bus after three measurements is presented in Table 3.

TABLE III

	Third Measurement Master BUS Signal "1"	
	BUS_{IN}	$A_2=BUS_{OUT}$
Device 0	1	0
Device 1	0	0
Device 2	0	0
Device 3	0	0
Device 4	0	1
Device 5	1	1
Device 6	1	1
Device 7	1	1
Device 8	1	0
Device 9	0	0
Device 10	0	0
Device 11	0	0
Device 12	0	1
Device 13	1	1
Device 14	1	1
Device 15	1	1

The value i in the measurement counter is then compared with the maximum value for i, i_{max}, step 212, to determine if the measurement sequence has been completed. If i does not equal i_{max}, the measurement counter is incremented by one, step 214, and the process continues with a next measurement.

In the fourth measurement, the master control module still holds its bus signal high, i.e., at a "1". The first device 112 reads this bus signal directly and determines if the content of its address register is zero, step 204. Since the content of the first three bits (A_2, A_1, A_0) of the address register of the first device is "000" from Tables 1-3, the first device 112 enables its inverter and the bus output 32a is the inverse of the bus input

30a, step 206, and is a "0". This output is the bus input 30b for device 114.

Device 114 then determines whether the content of its register is equal to zero. Since the content of the register of the second device is not zero, instead having a "001" in its address register's A_2, A_1, A_0 bits, the second device passes its bus input 30b through without inverting it, step 208; thus the bus output 32b of the second device 114 is a "0".

Device 116 then determines if the content of its register is equal to zero. However, because the third device 116 has a "010" in its register, the third device does not enable its inverter but passes its bus input 30c through without inverting it, step 208; thus its bus output 32c is a "0". This process continues in a similar fashion for each of the devices in the network.

Following the fourth measurement, each device again stores its own most recent output state in the next bit of its address register (A_3), step 210. A summary of the input and output states of all the devices on the bus after four measurements is presented in Table 4.

TABLE IV

	Fourth Measurement Master BUS Signal "1"	
	BUS_{IN}	$A_3=BUS_{OUT}$
Device 0	1	0
Device 1	0	0
Device 2	0	0
Device 3	0	0
Device 4	0	0
Device 5	0	0
Device 6	0	0
Device 7	0	0
Device 8	0	1
Device 9	1	1
Device 10	1	1
Device 11	1	1
Device 12	1	1
Device 13	1	1
Device 14	1	1
Device 15	1	1

The same sequence is performed until all $i=\log_2(n)$ measurements are made. When $i = i_{max}$, step 212, the auto-addressing mode is finished, step 216.

In each device, the content of its address register, stored in $A_i, \ldots A_2, A_1, A_0$, is the logical address of that device. Address zero is closest to the master control module, followed by addresses 1, 2, ..., n-1, with n-1 being the last device furthest from the master control module. This is demonstrated by combining the output columns of Tables 1-4, as shown in Table 5.

TABLE V

	A_3	A_2	A_1	A_0
Device 0	0	0	0	0
Device 1	0	0	0	1
Device 2	0	0	1	0
Device 3	0	0	1	1
Device 4	0	1	0	0
Device 5	0	1	0	1
Device 6	0	1	1	0
Device 7	0	1	1	1
Device 8	1	0	0	0
Device 9	1	0	0	1
Device 10	1	0	1	0
Device 11	1	0	1	1
Device 12	1	1	0	0
Device 13	1	1	0	1
Device 14	1	1	1	0
Device 15	1	1	1	1

Because the foregoing method is digital in nature, it does not suffer from variation in supply voltage, ambient temperature, sense resistance values, etc. This method also converges quickly, needing only $\log_2(n)$ measurements for n devices.

The foregoing descriptions and drawings should be considered as illustrative only of the principles of the invention. The invention may be configured in a variety ways and is not limited to the specific bus arrangement of the preferred embodiment. Numerous applications of the present invention will readily occur to those skilled in the art. Therefore, it is not desired to limit the invention to the specific examples disclosed or the exact construction and operation shown and described.

Rather, all suitable modifications and equivalents may be resorted to, falling within the scope of the invention.